(12) UK Patent Application (19) GB (11) 2 227 584 (13) A

(43) Date of A publication 01.08.1990

(21) Application No 8927269.4

(22) Date of filing 01.12.1989

(30) Priority data
(31) 8901932 (32) 28.01.1989 (33) GB

(71) Applicant
International Computers Limited

(Incorporated in the United Kingdom)

ICL House, Putney, London, SW15 1SW,
United Kingdom

(72) Inventors
Donald Bell
Charles Lomas

(74) Agent and/or Address for Service
D C Guyatt
STC Patents, West Road, Harlow, Essex, CM20 2SH,
United Kingdom

(51) INT CL^5
G06F 12/12

(52) UK CL (Edition K)
G4A AFL

(56) Documents cited
GB 2204973 A GB 2129585 A GB 2079996 A
US 4542453 A
Commodore 64 Programmer's Reference
Guide ,1st Edition, 1983, published by Commodore
Business Machines Inc, & Howard W. Sams &
Co. Inc., see esp. pp108-113

(58) Field of search
UK CL (Edition J) G4A AFL AMX
INT CL^4 G06F
Online database: WPI

(54) Computer control data modification system

(57) A data processing system has an erasable programmable read-only memory (EPROM) 22 for holding firmware, and an electrically-erasable programmable read-only memory (EEPROM) 23 for holding patch information specifying modifications to be made to the firmware. In operation, code is copied from the EPROM into a random-access memory (RAM) 21 and is then patched using the information in the EEPROM. The EEPROM can be accessed by a remote computer, 18, to write new patches into it. The EEPROM has separate areas for holding new and trusted patches. In operation, an attempt is made to modify the code using the new patches. If this is successful, the new patch area becomes the trusted patch area. Otherwise, the system reverts to the existing trusted patches, if any.

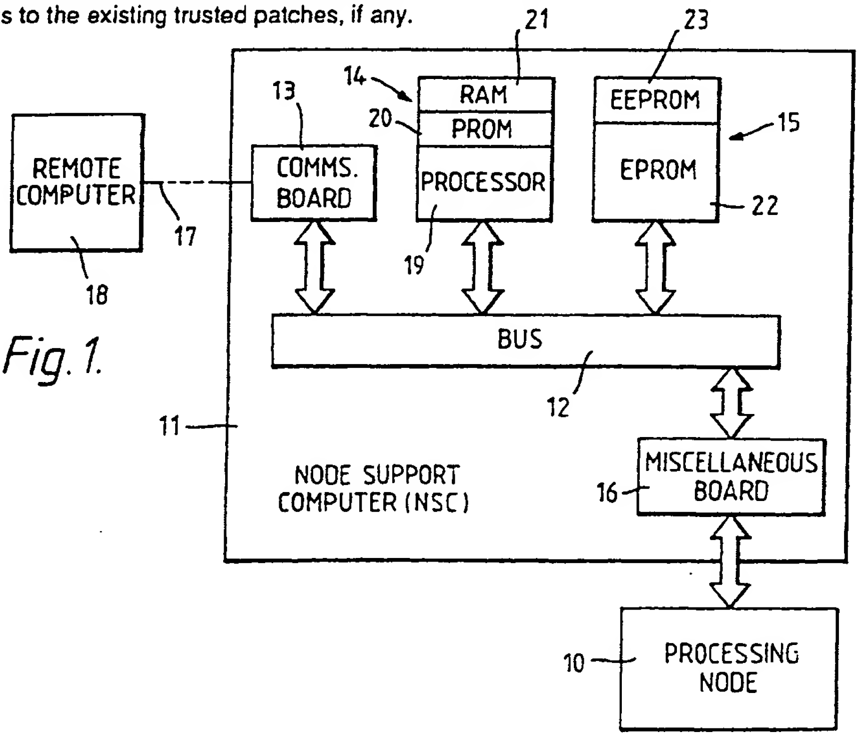


Fig. 1.

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

GB 2 227 584 A

Fig. 1.

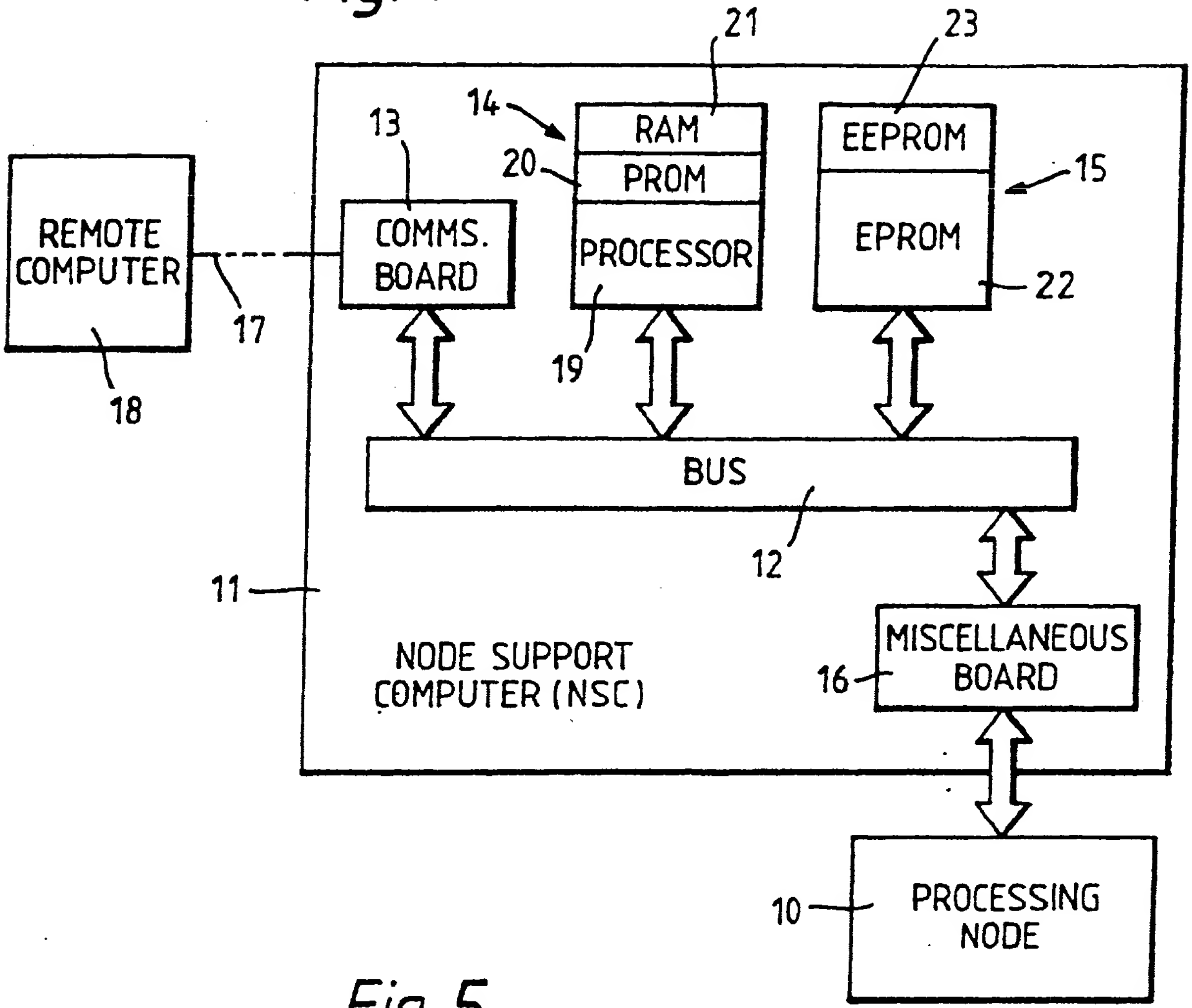

21
23
14
13
RAM
20
PROM
EEPROM
15
REMOTE COMPUTER
COMMS. BOARD
PROCESSOR
EPROM
22
17
19
18
BUS
12
11
MISCELLANEOUS BOARD
NODE SUPPORT COMPUTER (NSC)
16
10
PROCESSING NODE

Fig. 5.

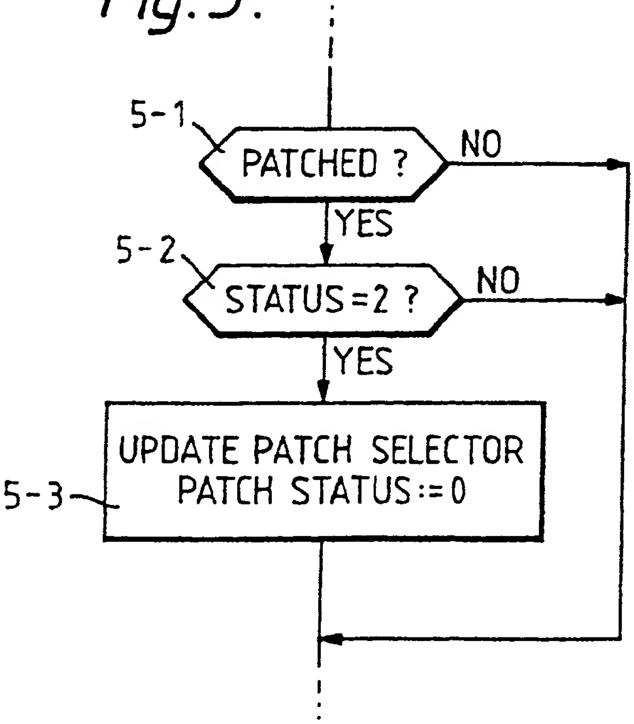

5-1
PATCHED ?
NO
YES
5-2
STATUS=2 ?
NO
YES
UPDATE PATCH SELECTOR
PATCH STATUS:=0
5-3

Fig. 2.

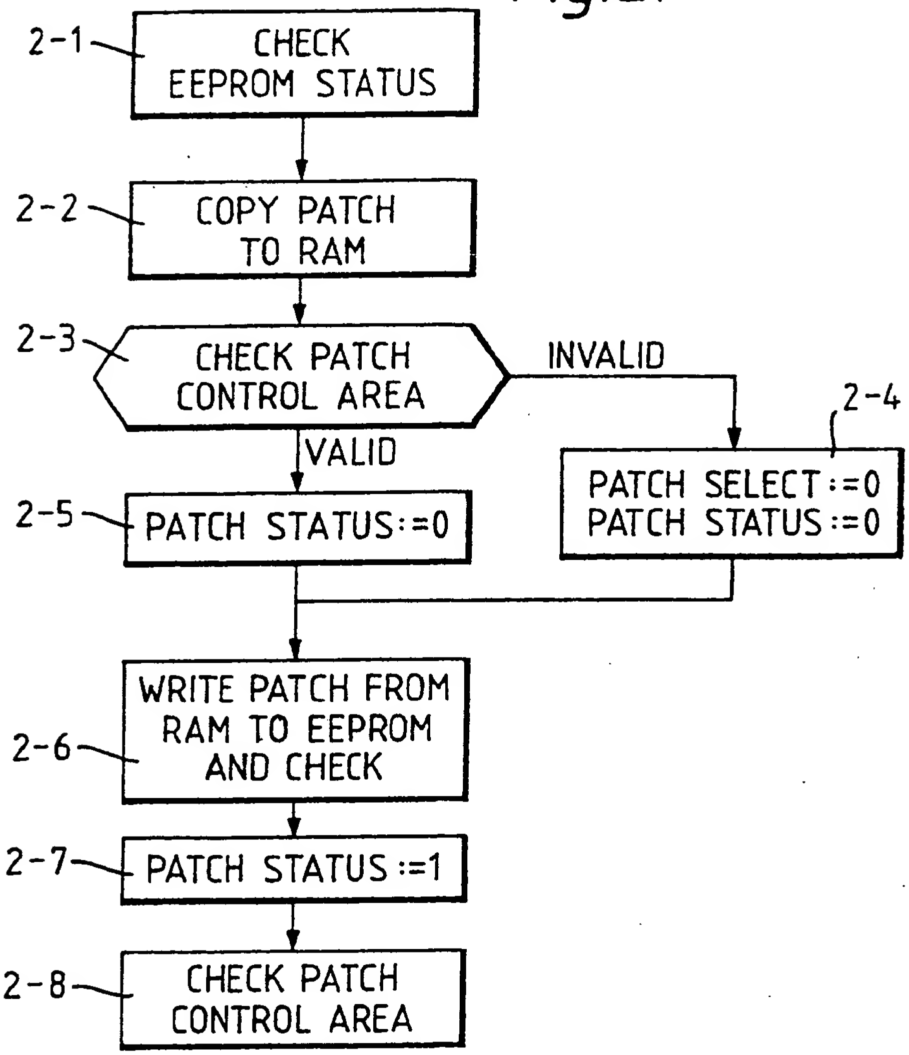

2-1
CHECK EEPROM STATUS
2-2
COPY PATCH TO RAM
2-3
CHECK PATCH CONTROL AREA
INVALID
2-4
VALID
2-5
PATCH STATUS:=0
PATCH SELECT:=0
PATCH STATUS:=0
2-6
WRITE PATCH FROM RAM TO EEPROM AND CHECK
2-7
PATCH STATUS:=1
2-8
CHECK PATCH CONTROL AREA

Fig. 3.

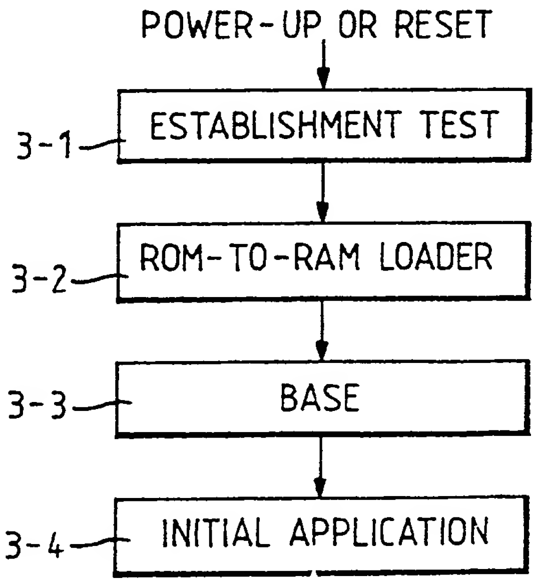
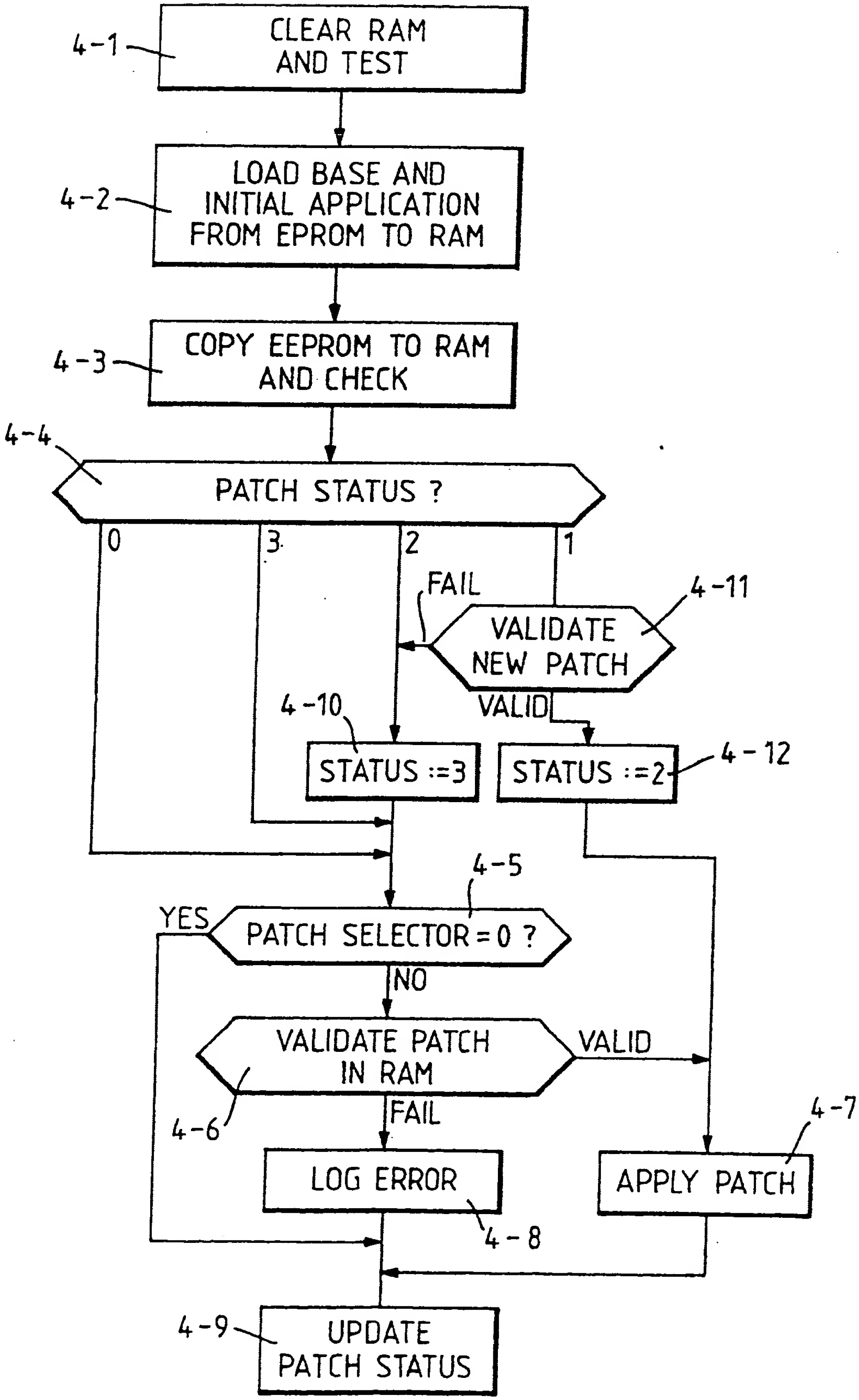

POWER-UP OR RESET
3-1
ESTABLISHMENT TEST
3-2
ROM-TO-RAM LOADER
3-3
BASE
3-4
INITIAL APPLICATION

Fig. 4.


4-1
CLEAR RAM AND TEST
4-2
LOAD BASE AND INITIAL APPLICATION FROM EPROM TO RAM
4-3
COPY EEPROM TO RAM AND CHECK
4-4
PATCH STATUS ?
0
3
2
1
FAIL
4-11
VALIDATE NEW PATCH
VALID
4-10
STATUS :=3
STATUS :=2
4-12
4-5
YES
PATCH SELECTOR = 0 ?
NO
VALIDATE PATCH IN RAM
VALID
4-6
FAIL
4-7
LOG ERROR
APPLY PATCH
4-8
4-9
UPDATE PATCH STATUS

C.1136

DATA PROCESSING SYSTEM

Background to the invention

This invention relates to data processing systems. More specifically, the invention is concerned with a data processing system including a data processor and a read-only memory (ROM) for holding control information for the processor. The control information may comprise, for example, firmware or other programs for controlling the operation of the processor.

In such a system, it may be necessary from time to time to update or correct the control information held in the ROM. One way of doing this is physically to replace the whole ROM with a new ROM. suitably programmed with the updated or corrected version of the control information. However, this is expensive and inconvenient, especially where a large ROM is involved. Also, in order to replace a ROM, it is usually necessary for skilled personnel to visit the user's site at which the system is located.

United States Patent No. 4 751 703 proposes a method of modifying programs held in a ROM without the necessity for physically replacing the ROM. The program code is divided into blocks, the first instruction of each block being held in a special read/write memory (RAM) while the remaining instructions are held in the ROM. A block of code can be patched by replacing the first instruction of that block by a jump to a replacement block. Thus, the program in the ROM can effectively be modified by modifying the information held in the RAM.

However, a problem with the above method is that it requires a virtual addressing scheme for addressing the code, since the first instruction of each block is physically separate from the rest of the block. Moreover, the above method only allows the code to be modified by replacing the code on a block-by-block basis.

The object of the present invention is to provide a way of updating or modifying information held in a ROM without the necessity for physical replacement of the ROM, while avoiding the problems mentioned above.

Summary of the invention

According to the invention there is provided a data processing system comprising:

(a) a data processor

(b) a read-only memory for holding control information for the data processor,

(c) a read/write random access memory,

(d) a further memory for holding modification information defining modifications to be applied to the control information,

(e) means for copying the control information from the read-only memory into the random-access memory, and

(f) means for using the modification information in the further memory to modify the copy of the control information in the random-access memory.

One data processing system in accordance with the present invention will now be described by way of example with reference to the accompanying drawings.

Brief description of the drawings

Figure 1 is a block diagram of a data processing system embodying the invention, the system including a node support computer (NSC).

Figure 2 is a flow chart showing actions performed by a remote computer to write patch information into the NSC.

Figure 3 is a flow chart showing operation of the NSC when it is powered up or reset.

Figure 4 is a flow chart showing a ROM-to-RAM loader program executed by the NSC.

Figure 5 is a flow chart showing part of an Initial Application program executed by the NSC.

Description of an embodiment of the invention

Referring to Figure 1, the data processing system comprises a main computer, referred to as a processing node 10, and a subsidiary computer referred to as a node support computer (NSC) 11. The purpose of the NSC is to assist with initial program load of the processing node, and to provide various diagnostic and monitoring facilities for the processing node. The structure of the processing node 10 forms no part of the present invention and so will not be described herein.

The NSC comprises an internal bus 12, which interconnects a communications board 13, a processor board 14, a ROM board 15, and a miscellaneous board 16.

The communications board 13 connects the NSC by way of a communications link 17 (e.g. a telephone line) to a remote computer 18. This allows the remote computer to perform diagnostic operations on the system and also, as will be described, to load patch information into the NSC.

The processor board 14 includes a conventional microprocessor 19, a 16Kbyte programmable read-only memory (PROM) 20, and a 1Mbyte dynamic random-access memory (RAM) 21. In operation, the microprocessor executes programs held either in the PROM 20 or in the RAM 21.

The ROM board 15 comprises a 2Mbyte erasable programmable read-only memory (EPROM) 22 and a 16Kbyte electrically erasable programmable read-only memory (EEPROM) 23. The EPROM holds programs (firmware) which in operation are loaded into the RAM on the processor board, for execution by the processor, as well as programs for the processing node 10. The EEPROM holds patch information indicating modifications to be made to the programs loaded into the RAM from the EPROM. The miscellaneous board 26 includes drive circuits for coupling the NSC to the processing node 10.

EEPROM contents

The EEPROM contains 16Kbytes which are addressed as segment number E800, byte offset 0000-7FFF (in hexadecimal notation). The contents of the EEPROM are as follows:-

byte offset	Contents
0000-07CF	reserved
07D0-07FF	control area
0800-2FFF	reserved
3000-57FF	patch area 1
5800-7FFF	patch area 2

Patch areas 1 and 2 are used to hold patch information indicating modifications to be made to the programs loaded from the EPROM into the RAM. In operation, either of these two patch areas can be designated the "trusted" patch area, and is used to hold tried and tested patch information. The other of the areas is designated the "new" patch area, and is used to hold new patch information that has not yet been tried.

The control area contains information about the status of the patch areas as follows:-

PATCH BASE ROM ID: This indicates the identity of the EPROM to which the patches are to be applied.

PATCH VALIDITY : This is set to the hexadecimal value 1A 2B 3C 4D if the other items in the control area are valid.

PATCH SELECTOR: This identifies the current "trusted" patch area (if any) and also, by implication, identifies the "new" patch area, as follows. If PATCH SECTOR = 0, there is no trusted patch information available, and patch area 1 is designated the new patch area. If PATCH SECTOR = 1, then patch area 1 is the trusted patch area and patch area 2 is the new patch area. If PATCH SELECTOR = 2, then patch area 2 is the trusted patch area and patch area 1 is the new patch area.

PATCH STATUS: This indicates the status pf the new patch, if any. If PATCH STATUS = 0, there is no new patch. If PATCH STATUS = 1, a new patch is ready to be tried. If PATCH STATUS = 2, a new patch is currently being tried. If PATCH STATUS = 3, a new patch has failed.

Each of the patch areas consist of a header, followed by a number of patch records of variable length. Each patch record comprises the following fields.

RECORD LENGTH: this indicates the number of bytes in the record, inclusive of this length field.

RAM SEGMENT ADDRESS and RAM BYTE OFFSET: these two fields specify the position within the RAM at which the patch data is to be written.

PATCH DATA: this is the patch data which is to be written into the RAM to modify the programs held in the RAM.

Each patch record can specify either:

(a) a modification to an existing byte (or contiguous sequence of existing bytes) in the RAM, or

(b) binary data to be written into a reserved area of the RAM.

Using these patch records, a program held in the RAM can be modified in two different ways.

(a) It can be modified by a straightforward byte-for-byte substitution, specified by one or more patch records.

(b) Alternatively, a patch record can be used to insert a new sequence of code in the reserved area of the RAM, while another patch record is used to plant a jump instruction to the start of the new sequence, at the appropriate location of the existing code.

Transfer of patch records

The patch records are generated by the remote computer 18, and are held in a patch file in that computer. When it is desired to update the firmware held in the NSC, the patch records are transferred from the remote computer 18, by way of the communications link 17, into the EEPROM in the NSC.

The procedure for transferring the patch records is shown in Figure 2.

(2-1) First, the remote computer sends a message to the NSC, requesting the status of the NSC. This contains, among other things, an indication of the status of the EEPROM. If the EEPROM is deemed to have failed, then an error message is generated, and the attempt to transfer the patch records is abandoned.

(2-2) Assuming that the EEPROM has not failed, the remote computer then accesses the NSC ROM VERSION, i.e. the version number of the EPROM currently installed in the NSC, and checks whether there are any patch records available in the patch file for that ROM version. If there are patch records available, the remote computer then writes these items into the RAM in the NSC.

The remote computer then reads the patch records back again, and compares them with the data that was written. If the comparison fails, an error message is generated, suggesting a possible communications problem, and the attempt to transfer the patch records is abandoned.

(2-3) Assuming that the comparison is successful, the remote computer now reads the current contents of the patch control area from the EEPROM in the NSC, and checks whether it is valid.

(2-4) If the contents of the patch control area are not valid, the PATCH SELECTOR field is set to 0 to indicate that there is no trusted patch information currently available in the EEPROM. The PATCH STATUS field is also set to 0 to indicate that there is, as yet, no new patch information in the EEPROM.

(2-5) If, on the other hand, the contents of the patch control area are valid, it is assumed that there is already some trusted patch information in the patch area indicated by the PATCH SELECTOR field. In this case, only the PATCH STATUS field is set to 0.

If, at steps 2-4 or 2-5, any of the writes to the control area of the EPROM fails, an error message is generated, indicating that the EEPROM has failed, and patching is abandoned.

(2-6) The patch records are now transferred from the RAM into the new patch area of the EEPROM, as indicated by the PATCH SELECTOR field. Again, if any of the writes to the EEPROM fails, an error message is generated and patching is abandoned.

The contents of the new patch area are then read back from the EEPROM and compared with the data that was written. If the comparison fails, an error message is generated, and patching is abandoned.

(2-7) Assuming that the comparison is successful, the PATCH STATUS field of the control area of the EEPROM is set to 1 to indicate that new patch information is now being tried.

(2-8) Finally, the whole of the patch control area is read back from the EEPROM and compared with the expected values. If the comparison fails, an attempt is made to set the PATCH VALIDITY field of the EEPROM to 0, an error message is output. and patching is abandoned.

Operation of NSC

Referring now to Figure 3, this shows the operation of the NSC when it is powered up or reset.

(3-1) The first action on power-up or reset is to run an establishment test program, which is resident in the PROM 20 on the processor board. This tests the basic facilities of the NSC.

(3-2) Assuming that the establishment test is successful, the NSC then performs a ROM-to-RAM loader program, which is also resident in the PROM 20. This loads BASE and INITIAL APPLICATION programs from the EPROM 22 into the RAM 21. It then applies the patches held in the EEPROM 23 to the RAM; that is, it modifies the programs in the RAM in accordance with the patch information in the EEPROM.

(3-3) At the end of the ROM-to-RAM loader, the NSC jumps to the start of the BASE program, which is now held in the RAM, modified by any patches. This is the basic operating program of the NSC.

(3-4) The BASE program calls the INITIAL APPLICATION program. This controls loading of firmware from the EPROM 22 into the processing node 10.

ROM-to-RAM loader

The ROM-to-RAM loader program referred to above will now be described in more detail with reference to Figure 4.

(4-1) The first action of the ROM-to-RAM loader is to clear the RAM 21 and to test it.

(4-2) The BASE and INITIAL APPLICATION programs are then loaded from the EPROM 22 into the RAM.

(4-3) The contents of the EEPROM 23 are then copied into a reserved area of the RAM. The copy is read back three times and compared with the contents of the EEPROM, to provide a stability check on the contents of the EEPROM. Also, the patch control area is checked to ensure that the PATCH VALIDITY field is set, and that other items in the control area have valid values.

(4-4) A branch is now made according to the value of the PATCH STATUS field of the patch control area.

If the PATCH STATUS is 0, there is no new patch available, and if PATCH STATUS is 3, there is a new patch, but it has failed for some reason. In either of these cases, the following actions are performed.

(4-5) If the PATCH SELECTOR field is non-zero, there is a trusted patch available, which can be used.

(4-6) The copy of the trusted patch in the RAM is checked to ensure that it is valid.

(4-7) If the patch is valid, it is applied to the programs previously loaded into the RAM. This involves reading each patch record in turn from the patch area, and writing the patch data in that record into the RAM at consecutive locations, starting from the byte specified by the RAM SEGMENT ADDRESS and RAM BYTE OFFSET.

(4-8) If it was found at step 4-7 above that the patch was invalid, an error is logged and the attempt to apply the patches is abandoned.

(4-9) If the patch status has been changed by the ROM-to-RAM loader, the EEPROM is accessed and the PATCH STATUS field is updated accordingly.

(4-10) If it was found at step 4-4 above that PATCH STATUS = 2, this indicates that a new patch has been tried, but the programs as modified by the new patch failed to run successfully. In this case, the PATCH STATUS is changed to 3 to indicate that the new patch has failed. The procedure then continues from step 4-5 as described above.

(4-11) If it was found at step 4-4 that PATCH STATUS = 1, this indicates that a new patch is ready to be tried. The copy of the new patch in the RAM is checked to ensure that it is valid.

(4-12) If the new patch is valid, then PATCH STATUS is set to 2 to indicate that a new patch is now being tried. The procedure then continues from step 4-7 as described above, using the designated new patch area to patch the RAM. If, on the other hand, the new patch is found to be invalid, the procedure continues from step 4-10 as described above.

INITIAL APPLICATION.

As described above, after the ROM-to-RAM loader program, the BASE and INITIAL APPLICATION programs are run.

At some convenient point in the execution of the INITIAL APPLICATION program, the actions shown in Figure 5 are performed.

(5-1) A test is made to determine whether or not the program has been patched.

(5-2) If so, the PATCH STATUS field is examined.

(5-3) If PATCH STATUS = 2, this indicates that a new patch is currently being tried. Since the INITIAL APPLICATION program is now running successfully with the new patch in place, it is assumed that the new patch is successful. Hence, the EEPROM is accessed and the PATCH SELECTOR is updated, to indicate that this new patch is now the trusted patch, and the PATCH STATUS is set to 0 to indicate that there is now no new patch. The program then continues running normally.

If either of the write accesses to the EEPROM fails, the NSC is reset, so that it will restart with the establishment test as shown in Figure 3.

Conclusion

In summary, it can be seen that the above sequences allow patch information to be written into the EEPROM, and then to be applied to the programs the next time the NSC is reset or powered up.

If the newly applied patch information corrupts the BASE or INITIAL APPLICATION programs to such an extent that the latter fails to get established, then at the next reset or power-up the new patch will be designated as "failed". The system will therefore revert to any existing trusted patches or, if there are no trusted patched, will proceed without any patches being applied.

CLAIMS

1. A data processing system comprising:-

(a) a data processor,

(b) a read-only memory for holding control information for the data processor,

(c) a read/write random-access memory,

(d) a further memory for holding modification information defining modifications to be applied to the control information,

(e) means for copying the control information from the read-only memory into the random-access memory, and

(f) means for using the modification information in the further memory to modify the copy of the control information in the random-access memory.

2. A system according to Claim 1 wherein the further memory is an electrically erasable programmable read-only memory (EEPROM).

3. A system according to Claim 1 or 2 wherein said further memory is accessible by a remote computer to allow the remote computer to write modification information into the further memory.

4. A system according to any preceding claim wherein said control information comprises a program for controlling the operation of the data processor.

5. A data processing system according to any preceding claim wherein said further memory comprises at least two areas, one of said areas being designated as a trusted area for holding modification information that has been successfully tried, and another of said areas being designated as a new area for holding modification information that has not yet been tried.

6. A data processing system substantially as hereinbefore described with reference to the accompanying drawings.

7. A method of operating a data processing system comprising:-

(a) storing control information in a read-only memory,

(b) storing modification information in a further memory,

(c) copying the control information from the read-only memory into a read/write random-access memory, and

(d) using the modification information in the further memory to modify the copy of the control information in the random-access memory.

8. A method according to Claim 7, further comprising the steps:-

(a) dividing the further memory into at least two areas,

(b) designating one area as a trusted area for holding modification information that has been successfully tried, and

(c) designating the other area as a new area for holding modification information that has not yet been tried.

9. A method according to Claim 8 wherein the step of using the modification information comprises checking the information in the new area and, if the checking is satisfactory:

(a) using the modification information in the new area to modify the control information in the random-access memory,

(b) operating the data processing system using the modified control information, and

(c) if the data processing system operates successfully using the modified control information, re-designating the new area as the trusted area.

10. A method according to Claim 9 wherein, if the data processing system does not operate successfully using the modified control information:

(a) the control information from the read-only memory is copied again into the random-access memory, and

(b) the modification information (if any) in the trusted area is used to modify the control information in the random-access memory.

11. A method according to any one of claims 7 to 10 including the step of writing modification information into the further memory from a remote computer by way of a communications link.

12. A method of operating a data processing system substantially as hereinbefore described with reference to the accompanying drawings.

THIS PAGE BLANK (USPTO)